


# OPERATÖRLER

## ▼ Operatör nedir?

Belirli bir sorumluluğu/ işi/ operasyonu üstlenen sembolik ya da metinsel yapılardır. Bizim yerimize o sorumluluğu inşa eder.

## ▼ Operatörlerin okur yazarlığı



Operatorler; genellikle iki deger arasında matematiksel, mantiksal yahut farkli bir islemsel gorev/operasyon yapan yapılardır.

Operatorler genellikle yaptıkları işlem neticesinde bir sonuc donerler.

Operatorleri kullanırken geriye donus degerlerine dikkat edilmesi gerekmektedir.

## ▼ Aritmetik operatörler

Aritmetik operatörler, iki farklı değer üzerinde işlem yapan operatörler oldukları için işlem neticesinde geriye UYGUN türde sonuç dönerler.

int-int → int döner (Aynı türler arasındaki işlem döndürüldüğünde aynısı döner)

int-double → double döner

byte -int → int (Hangi değer daha büyükse onu döndürür.)

### !!!! İSTİSNA

İki byte arasında yapılan tüm işlemlerde INT döner. Bu böyle kabul edilmiştir.

#### ▼ Karşılaştırma operatörleri

Geriye döndüreceği değer true ya da false'dur.

<, >, ==, <=, >=

#### ▼ Mantıksal operatörler

- Tüm şartları değerlendirip, kendine göre değer döndüren operatörlerdir.
  - && → ve operatörü : tüm şartların yerine getirilmesini ister. Hem patates hem köfte.
  - || → veya operatörü : şartlardan en az bir tanesinin yerine getirilmiş olsun. Ya patates ya köfte, ya da ikisi de
  - ^ → ya da operatörü : ikisi aynı anda da gelemez, ikisi gelmezse de olmaz. Kesinlikle 1 tanesinin yerine getirilmesini ister.

#### ▼ Arttırma azaltma operatörleri

Yalnızca 1 tane arttırabilirsin. ++, --

Noramlde `i = i + 5` yazılıp, `cw(i)` yazmamız gerekirdi ama bu operatörler ile `cw(i++)` apmamız yeterli olur.

#### **++i ile i++ arasındaki fark nedir?**

`++i` → öncelikle i değerini arttırır sonra i yi döndürür.

`i++` → Öncelikle i değerini döndürür sonra i yi arttırır.

```
#region Arttırma Azaltma Operatörleri
// ++
int i = 5;
//i++;
//++i;
//Console.WriteLine(i);
```

```
Console.WriteLine(i++); // Çıktı : 5 | Bellek : 6
Console.WriteLine(++i); // Çıktı : 7 | Bellek : 7
```

Micro  
5  
7  
C:\User  
e 0.  
To auto

```
#region Arttırma Azaltma Operatörleri
```

Örnek 1

```
#region Örnek 2
```

```
int a = 5;
int b = a++;
Console.WriteLine(a);
Console.WriteLine(b);
#endregion
#endregion
```

STACK

(int) a = 6

(int) b = 5

#### ▼ Üzerine ekleme , yığma operatörleri

- 1 den fazla değer eklemek istediğimizde yapabiliriz örneğin;
  - $i+=3 \rightarrow i = i+3$  ile aynı şey
  - $i-=3$
  - $i*=3$
  - $i/=3$
  - $i\%=3$

#### ▼ Metinsel ifadelerde operatörler

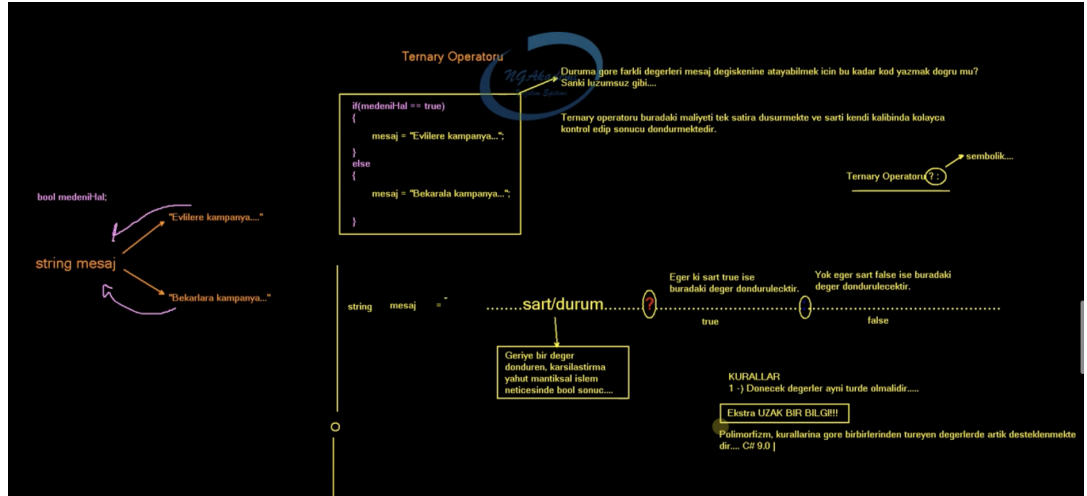
- Metinsel ifadeler +operatörü ile birleştirilebilirler.



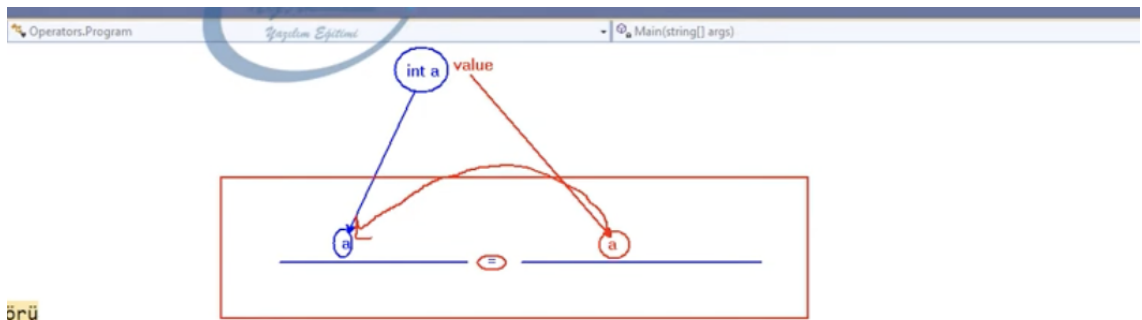
- == operatörü ile bool sonuç döndürerek eşit olup olmadığına bakabiliriz
- += operatörü ile a=ahmet b = behmet değerlerini toplayıp a ya atayabiliriz.  
a artık ahmetbehmet olur.  
a=a+b şeklinde → a+=b;
- != eşit değil mi operatörü

#### ▼ Diğer operatörler

- ! operatörü
  - Olumsuzluk ifade eder. !true → false olur.
  - Eşit deillik durumu.  
!=
- Ternary operatörü
  - Şarta bağlı değer döndüren operatördür.
  - Bir değişkene / methoda/property'e değer atarken, eğer ki bu değer şarta göre fark edecekse satır bazlı / tek satırda şart kontrolunu yaparak duruma göre değeri döndürmemizi sağlayan kalıpsal operatördür.



- Atama operatörü (Assign)




\*\*\* İleride(Referans türlü değişkenlerde) atama operatörünün sorumluluğu değişip referans etme operatörü olduğunu konuşacağız....|

- Member Access- Üye erişimi operatörü
  - Nokta (.) 'dır.
  - int i = 5;
  - i.ToString () gibi gibi alt üyelerine erişmemizi sağlayan operatörüdür.
- Cast operatörü (Detaylıca inceledik önceki derslerde)
  - Boxing → unboxing
  - Bilinçli tür dönüşümü

- Char → int | int → char (ascii)
- sizeof operatörü
  - Verilen türün bellekte kaç byte'lık yer kapladığını integer olarak geriye döndürür.
  - C.W ( "int: " + sizeof (int));
- typeof Operatörü
  - Verilen türün / değerın type'ını/türünü getirir.

```
0 references
static void Main(string[] args)
{
    #region typeof Operatörü
    //typeof operatörü verilen türün/değerın type'ını/türünü getirir.
    //O tür ile ilgili bilgileri edinmek için kullanılan bir operatördür.
    //İleride (ileri düzey programlamada) reflection dediğimiz bir konuda elimizdeki bir türün reflectionına girmek için kullanıldığını
    göreceğiz.
    Type t = typeof(int); //int türüne ait tüm bilgiler burada t değişkenine atanmıştır.
    Console.WriteLine(t.Name);
    Console.WriteLine(t.IsPrimitive);
    Console.WriteLine(t.IsClass);
    Console.WriteLine(t.IsValueType);
    #endregion
}
```



```
int32
true
false
true
```

- Default operatörü
  - Her tür için yazılım tarafından tanımlanmış bir varsayılan değer demektir. Belirtilen türün default değerini döndürür.

```
#region default Operatörü
//Belirtilen türün default değerini döndüren operatördür...
//Default değer ne demektir? Default değerler, her tür için yazılım tarafından tanımlanmış bir varsayılan değer demektir.
//keyisel = 0
//bool = false
//string = null
//char = '\0'
//references = null

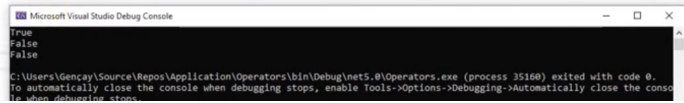
Console.WriteLine(Default(decimal));
Console.WriteLine(Default(string));
Console.WriteLine(Default(Program));
Console.WriteLine(Default(short));
Console.WriteLine(Default(byte));
```



```
0
,
,
,
,
```

- is operatörü

```
0 references
static void Main(string[] args)
{
    #region is Operatörü
    //Boxing'e tabi tutulmuş bir değerin öz türünü öğrenebilmek/check edebilmek/kontrol edebilmek için kullanılan bir operatördür.
    //is operatörü denetleme neticesinde durumu bool yani true ya da false olarak döndürecektir.
    object x = true; //Boxing
    Console.WriteLine(x is bool);
    Console.WriteLine(x is int);
    Console.WriteLine(x is Program);
    #endregion
}
```



```
true
false
false
```

- is null operatörü

```
#region is null Operatörü
//Bir değişkenin değerinin null olup olmasını kontrol eden ve sonuç olarak geriye bool türde bir değer döndüren operatördür.
string a = null;
Console.WriteLine(a is null);

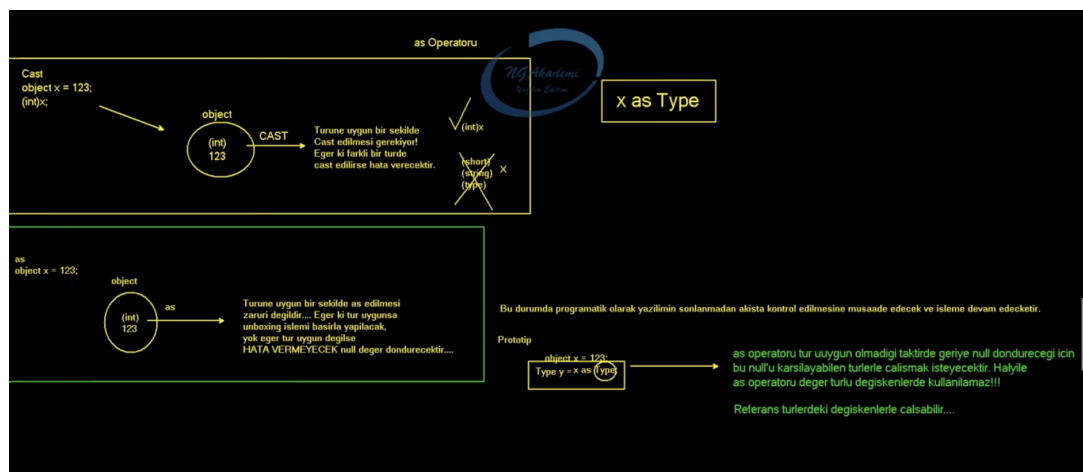
//is null operatörünü sadece null olabilen türlerde kullanabilmekteyiz.
int b = 123;
Console.WriteLine(b is null);
#endregion
```

- is not null operatörü

```
0 references
static void Main(string[] args)
{
    #region is not null Operatörü
    //Eliizdeki değerin null olup olmamasıyla ilgilenmekte ve geriye bool sonuç döndürmektedir.
    string a = null;
    Console.WriteLine(a is not null);

    //Sadece null alabilen türlerde kullanılabılır...
    #endregion
}
```

- as operatörü
  - cast operatörüne alternatiftir.



- nullable operatörü

```
#region ?(Nullable) Operatörü
//C# prog. dilinde değer tür. değişkenler normal şartlarda null değer alamazlar(Not nullable)
//Bir değer türlü değişkenin null değer alabilmesi için(yani nullable olabilmesi için) ? operatörünün kullanılması gerekmektedir.
int? a = null;
bool? b = null;
#endregion
```

Prototip : `degisken_turu? degisken_adi;`  
 nullable

Bu formatta tanımlanan değişkenler null değer alabilirler

```
#region ?(Nullable) Operatörü
//C# prog. dilinde değer tür. değişkenler normal şartlarda null değer alamazlar(Not nullable)
//Bir değer türlü değişkenin null değer alabilmesi için(yani nullable olabilmesi için) ? operatörünün kullanılması gerekmektedir.
int? a = null;
bool? b = null;

Console.WriteLine(a is null);

//Bir değer türlü değişken nullable yapıldıysa eğer; is null, is not null, as gibi null ile çalışan operatörleri üzerinde kullanabiliriz.

#endregion as Örneklendirme
object x = 123;
int? y = x as int?;
#endregion
#endregion
```

- null coalescing operatörü

```
#region ??(Null-Coalescing) Operatörü
//Elimizdeki değere null değeri verilmeden farklı bir değeri göndermemizi sağlayan operatördür
#endregion
```

string a = null;

a ?? "Merhaba"

a değişkeninin değeri null değilse a'nın değerini yazdır, yok eğer null ise "Merhaba" yazdır....

- sol ve sağdaki türler aynı olmalıdır. a ?? b olamaz

- null coalescing assignment

- ??= x in değeri null ise merhaba yazdır. x in değerini direkt yazdır.
  - cw( x ??= "Merhaba")